Análisis de capacidad

Pruebas:

Las pruebas realizadas para esta iteración fueron tomadas y extendidas de la iteración anterior. En este caso utilizamos 8 hilos de ejecución cada uno realizando 5 pruebas en un tiempo distribuido entre 15 a 40 segundos. Para ello utilizamos el software para pruebas de rendimiento JMeter que hacía peticiones al proyecto alojado en la nube AWS. La razón por la cual se disminuyó el número de pruebas por hilo se debe a 2 factores:

- El tiempo de procesamiento del worker es muy alto con tantas tareas encoladas en las pruebas donde el sistema no escala, por lo cual no es práctico
- A mayor cantidad de pruebas no se obtienen valores muy diferentes a los obtenidos en esta prueba. Contrario a esto, las ejecuciones al tener un tiempo mayor de ejecución y ser ejecutadas a un sistema fuera de la nube, pueden obtener ruidos de diferentes factores como red, el sistema donde ejecutan las pruebas o el sistema donde se ejecuta la solución desarrollada.

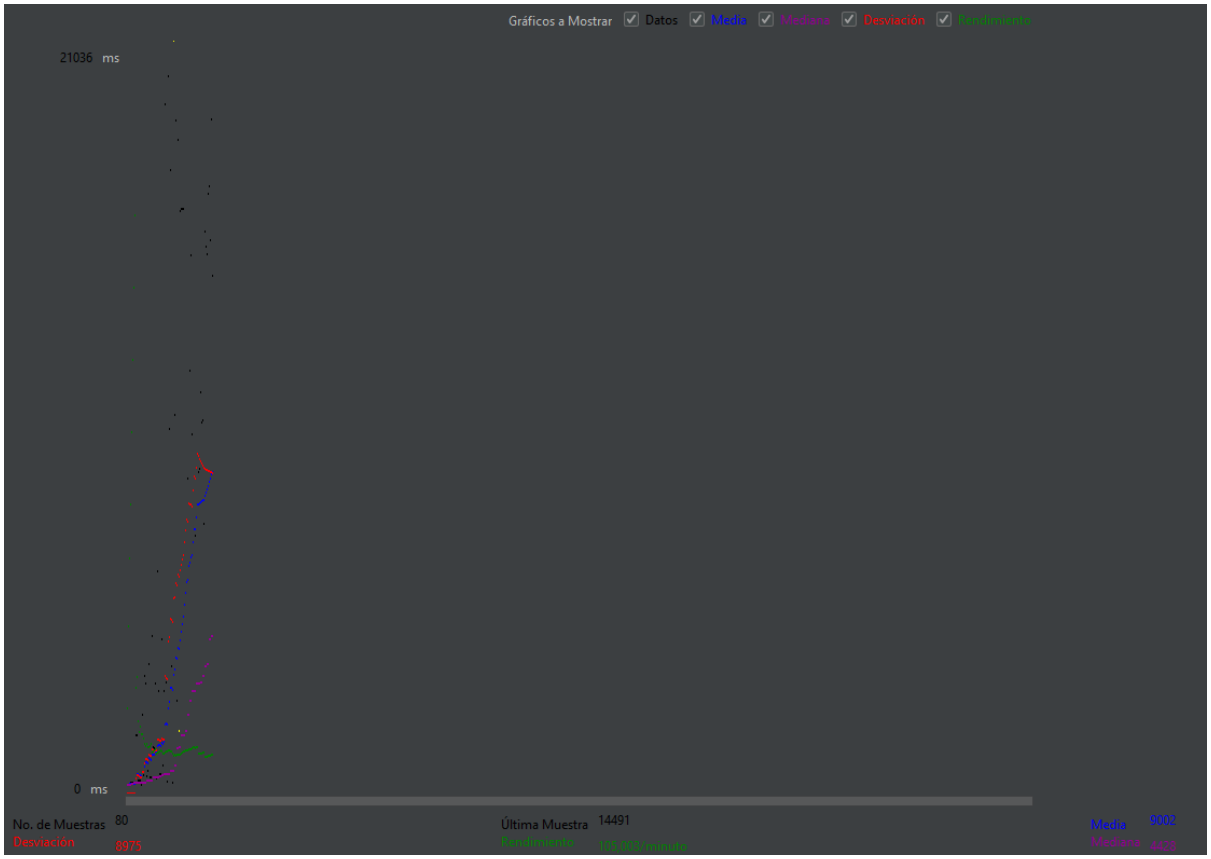
Con el propósito de evidenciar los beneficios o efectos del auto escalamiento y el uso de balanceadores de carga, se hicieron 2 pruebas aisladas:

- Caso 1: Se utiliza una sola instancia y se hacen peticiones a la API a través del balanceador de cara. En el caso del worker, recibe los videos que se encolan una vez se hacen las peticiones a través de la API.
- Caso 2: Inicialmente se utiliza una sola instancia y se hacen peticiones a la API a través del balanceador de carga. En dado caso de que la máquina virtual donde esta alojada la API llega a un consumo de CPU del 40%, se escala horizontalmente para que el balanceador de carga distribuya las peticiones en 2 máquinas virtuales. En el caso del worker, al llegar la cola a 10 peticiones se escala horizontalmente para distribuir las peticiones, Esto hasta llegar a un máximo de 3 máquinas virtuales con el worker.

Inicialmente se presentarán los resultados presentados por el software JMeter.

Caso 1:

En este caso vemos como la curva de rendimiento en las primeras peticiones se recibe una respuesta favorable, pero rápidamente disminuye y se estabiliza con un rendimiento no muy favorable de 105,003/minuto.



Al observar la tabla de resultados de Jmeter para el caso donde no hay escalamiento, se encuentra que hubo un error del 2.5%, esto representa error en una ejecución completa. Este comportamiento en este tipo de pruebas es difícil de definir si es debido a un error en el sistema a evaluar o en la red en donde están siendo ejecutadas las pruebas.

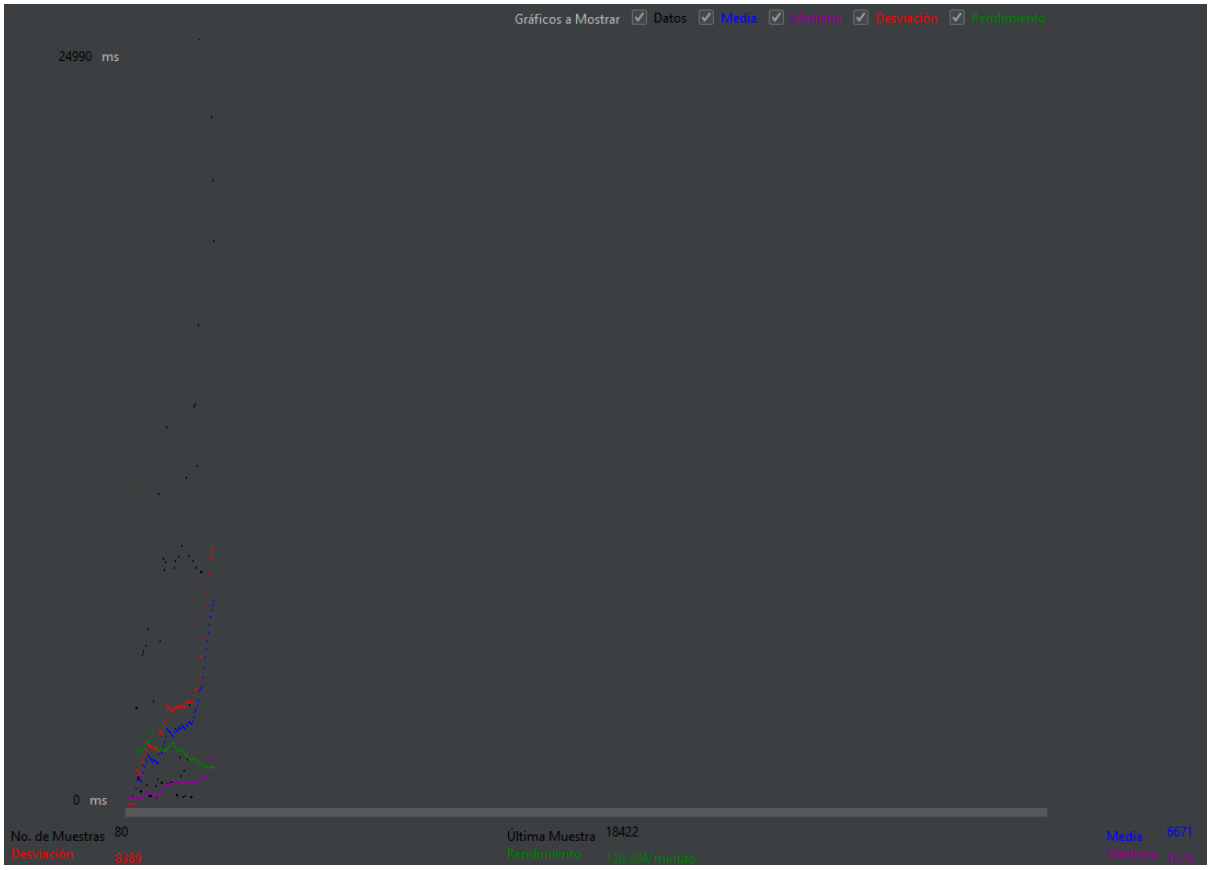
Fuera de este problema, todo el proceso fue ejecutado de forma correcta, pero con un rendimiento claramente menor respecto al sistema que tiene la capacidad de escalar.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Login	40	6915	264	28341	8928,25	2,50%	1,3/sec	0,68	0,33	544,9
Create_task	40	11088	1645	31484	8526,10	2,50%	52,8/min	0,21	2578,61	247,4
Total	80	9002	264	31484	8975,34	2,50%	1,8/sec	0,68	2563,94	396,2

Etiqueta	# Muestras	Media	Min	Max	Desv. Estandar	% Error	Rendimiento (s)	Kb/sec	Sent KB/sec	Media Bytes
Login	40	6915	264	28341	8928.25	2.5	1.3	0.68	0.33	544.9
Create_task	40	11088	1645	31484	8526.1	2.5	52.8	0.21	2578.6	247.4
Total	80	9002	264	31484	8975.34	2.5	1.8	0.68	2563.9	396.2

Caso 2:

En este caso se aprecia como la curva de rendimiento se estabiliza en las primeras peticiones en un nivel cercano a 120,304/minuto y aumenta probablemente al momento en el que escala horizontalmente el sistema. Eventualmente el rendimiento se ve afectado, pero mantiene un mejor rendimiento en comparación al caso inicial.



Observando la tabla de resultados, se evidencia una ejecución correcta de todas las pruebas e hilos. Un rendimiento mucho mejor de las API probadas, principalmente de la API que crea la tarea que es encolada y eventualmente procesada por el worker. Por último, se observa mínimos menores respecto a la prueba donde el sistema no tiene la funcionalidad de escalar.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Login	40	542	230	1574	337,34	0,00%	1,6/sec	0,78	0,44	486,8
Create_task	40	12799	3161	27617	8095,00	0,00%	1,0/sec	0,25	2954,48	249,4
Total	80	6671	230	27617	8389,41	0,00%	2,0/sec	0,72	2937,57	368,1

Etiqueta	# Muestras	Media	Min	Max	Desv. Estandar	% Error	Rendimiento (s)	Kb/sec	Sent KB/sec	Media Bytes
Login	40	542	230	1574	337.34	0	1.6	0.78	0.44	486.8
Create_task	40	12799	3161	27617	8095	0	1	0.24	2954.48	249.4
Total	80	6671	230	27617	8389.41	0	2	0.72	2937.57	368.1

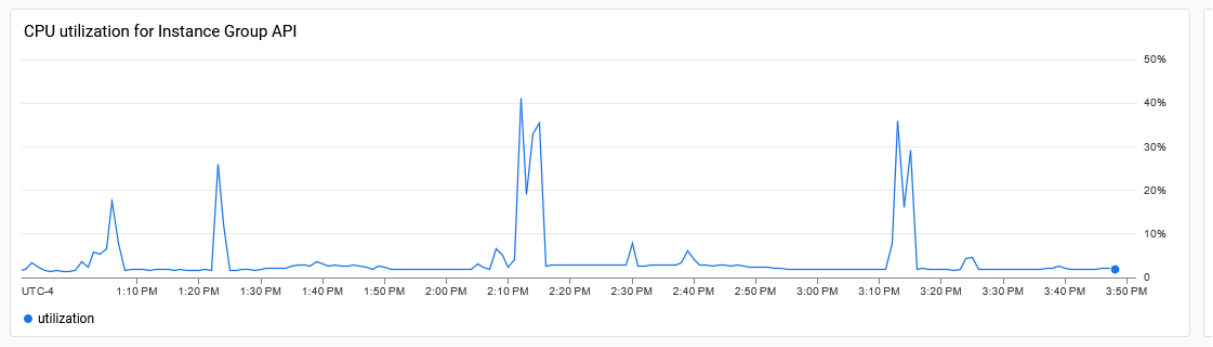
Análisis de resultados del sistema:

Todas las gráficas presentadas en este numeral tienen la misma periodicidad de tiempo y se puede evidenciar pruebas internas del equipo que no tienen relación con las pruebas de este documento y las pruebas relacionadas directamente con los dos casos evaluados. las pruebas definitivas presentadas en este documento hacen referencia a los tiempos entre las 2:00 PM – 2:20 PM para el caso 1 y entre las 3:10 PM y 3:20 PM para el caso 2. Se debe tener en cuenta que el sistema funciona en la región de Iowa con un

time zone de UTC-4 y el equipo que está desarrollando las pruebas está en Colombia con UTC-5.

Análisis de comportamiento API:

La siguiente gráfica muestra el comportamiento de CPU del grupo de recursos donde esta alojada la API al momento de hacer las pruebas.

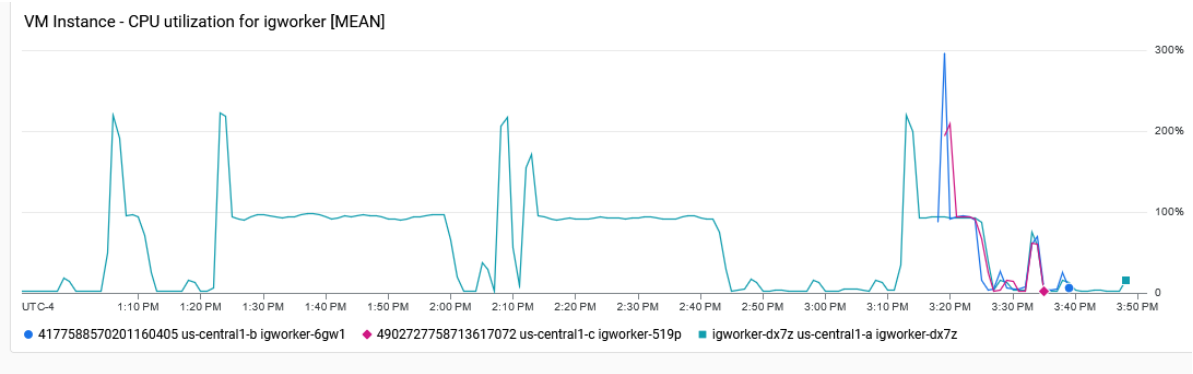


El consumo de CPU en el caso 1 tiene un pico mayor al 40% y continúa trabajando por el periodo de tiempo de la prueba consumiendo más del 20% de la capacidad. En el caso 2 las pruebas tienen una duración menor, cerca de la mitad del tiempo en comparación al caso 1. Adicional a esto, vemos como la condición de escalamiento se cumple y el grupo de instancias con la API escala al acercarse a un consumo del 40% de la CPU.

La gráfica a continuación permite evidenciar el número de instancias que se utilizan para el periodo de tiempo de cada prueba. El caso en donde se evidencia 2 instancias entre las 2:15 PM y las 2:30 PM se debe a una prueba interna del equipo, no afectan las pruebas del caso 1 y no tienen nada que ver para esto.

Análisis de comportamiento worker:

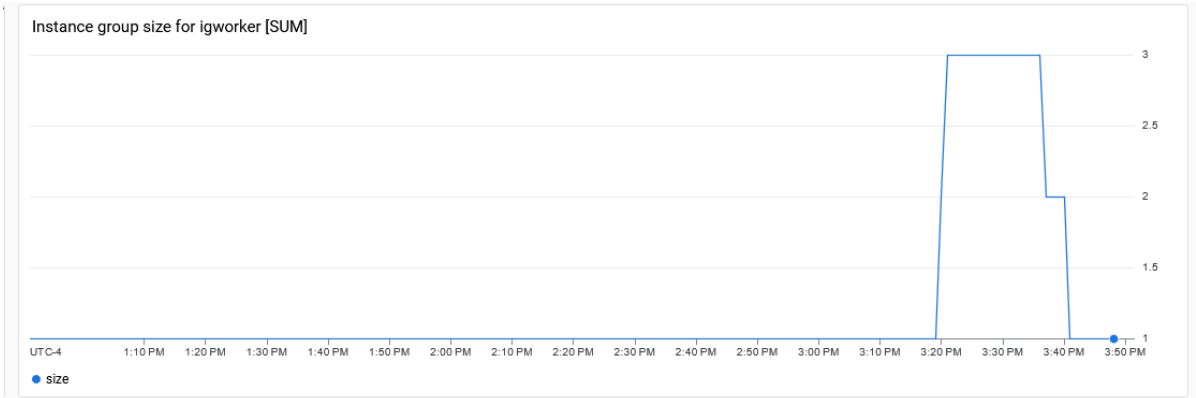
La siguiente gráfica y teniendo en cuenta solo los tiempos de evaluación mencionados anteriormente se obtienen los siguientes resultados de consumo de CPU para el worker:



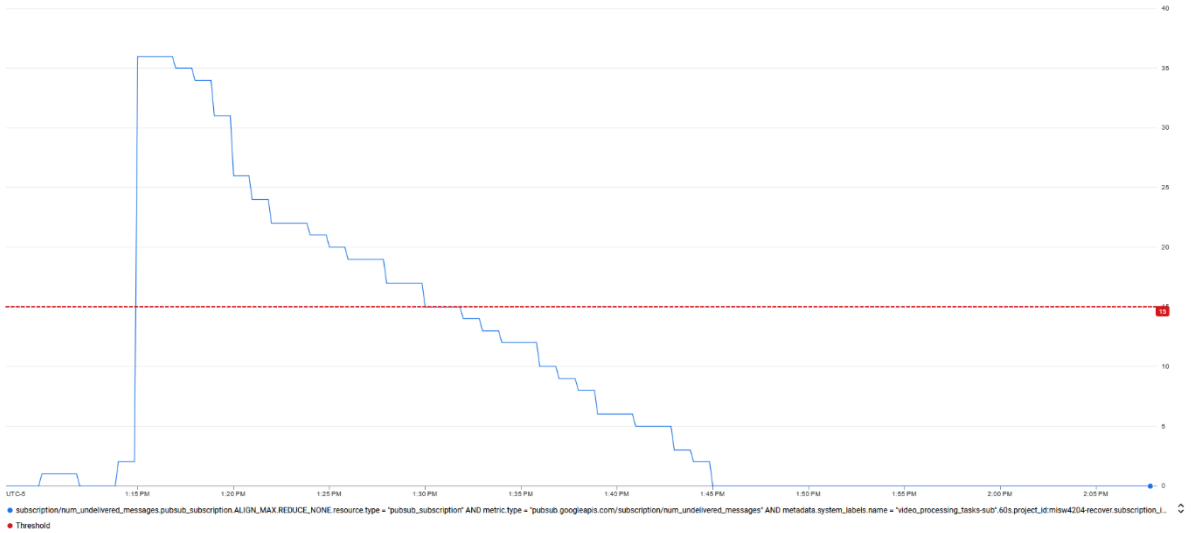
Se evidencia como en el caso 1 hay picos de consumo de CPU del 200%, esto debido al ruido o recursos que requiere el sistema para iniciar el procesamiento de videos. Después de esto, se evidencia un consumo cercano al 100% de los recursos. Al momento de hacer las pruebas, el equipo evidencio como la interfaz gráfica del sistema que muestra las colas y el tiempo de procesamiento fallaba, sin embargo, el sistema nunca tuvo periodos fuera de funcionamiento, dejo tareas por desencolar o sin procesar.

Para el caso 2 vemos como el proceso inicia como el caso anterior y al momento de tener más de 10 tareas enciende otra máquina virtual para cumplir con las condiciones estipuladas inicialmente. El comportamiento de las máquinas virtuales generadas por el proceso de escalamiento es igual que las mencionadas anteriormente. Al final, todas las colas tienen un porcentaje de consumo de CPU también cercano al 100% pero un tiempo de procesamiento mucho menor, favoreciendo el funcionamiento del sistema y dando una mejor experiencia para el usuario.

En la próxima gráfica, se puede evidenciar el mismo periodo de tiempo de la gráfica anterior. En esta gráfica se evidencia el número de instancias del worker, allí es posible ver como el proceso de escalamiento empieza aumentando el número de instancias hasta tener 3 instancias como es definido. Adicional, al final de la prueba cuando una instancia deja de ser utilizada se apaga y por un corto periodo de tiempo se utilizan 2 instancias.



Yendo al detalle del procesamiento del worker, Para el caso 1 worker tardo en procesar las 39 peticiones que recibió efectivamente en unos 32 minutos. Esto se puede evidenciar en la siguiente gráfica junto con la distribución de procesamiento de tareas:



Para el caso 2, los 3 workers utilizados para procesar las 40 peticiones generadas tardaron aproximadamente 17 minutos. Casi la mitad del tiempo respecto a las pruebas con el sistema sin escalamiento, esto y la distribución de procesamiento de tareas se evidencia en la siguiente gráfica:

De las dos pruebas desarrolladas, es importante mencionar que el procesamiento del worker disminuye conforme aumenta el número de peticiones. Además de esto, GCP hace revalidaciones que tardan aproximadamente 5 minutos antes de escalar la instancia.